Approximating Directed Steiner Problems via Tree Embedding

Bundit Laekhanukit[*][†]

February 29, 2016

Abstract

Directed Steiner problems are fundamental problems in Combinatorial Optimization and Theoretical Computer Science. An important problem in this genre is the *k-edge connected directed Steiner tree* (k-DST) problem. In this problem, we are given a directed graph G on n vertices with edge-costs, a root vertex r, a set of h terminals T and an integer k. The goal is to find a min-cost subgraph $H \subseteq G$ that connects r to each terminal $t \in T$ by k edge-disjoint r,t-paths. This problem includes as special cases the well-known *directed Steiner tree* (DST) problem (the case $k = 1$) and the *group Steiner tree* (GST) problem. Despite having been studied and mentioned many times in literature, e.g., by Feldman et al. [SODA'09, JCSS'12], by Cheriyan et al. [SODA'12, TALG'14] and by Laekhanukit [SODA'14], there was no known non-trivial approximation algorithm for k-DST for $k \geq 2$ even in the special case that an input graph is directed acyclic and has a constant number of layers. If an input graph is not acyclic, the complexity status of k-DST is not known even for a very strict special case that $k = 2$ and $|T| = 2$.

In this paper, we make a progress toward developing a non-trivial approximation algorithm for k-DST. We present an $O(D \cdot k^{D-1} \cdot \log n)$-approximation algorithm for k-DST on directed acyclic graphs (DAGs) with D layers, which can be extended to a special case of k-DST on "general graphs" when an instance has a *D-shallow* optimal solution, i.e., there exist k edge-disjoint r,t-paths, each of length at most D, for every terminal $t \in T$. For the case $k = 1$ (DST), our algorithm yields an approximation ratio of $O(D \log h)$, thus implying an $O(\log^3 h)$-approximation algorithm for DST that runs in quasi-polynomial-time (due to the height-reduction of Zelikovsky [Algorithmica'97]). Our algorithm is based on an LP-formulation that allows us to embed a solution to a tree-instance of GST, which does not preserve connectivity. We show, however, that one can randomly extract a solution of k-DST from the tree-instance of GST.

Our algorithm is almost tight when k, D are constants since the case that $k = 1$ and $D = 3$ is NP-hard to approximate to within a factor of $O(\log h)$, and our algorithm archives the same approximation ratio for this special case. We also remark that the $k^{1/4-\epsilon}$-hardness instance of k-DST is a DAG with 6 layers, and our algorithm gives $O(k^5 \log n)$-approximation for this special case. Consequently, as our algorithm works for general graphs, we obtain an $O(D \cdot k^{D-1} \cdot \log n)$-approximation algorithm for a D-shallow instance of the *k edge-connected directed Steiner subgraph* problem, where we wish to connect every pair of terminals by k edge-disjoint paths.

[*]The Weizmann Institute of Science, Israel, email:`bundit.laekhanukit@weizmann.ac.il`.

[†]The work was partly done while the author was at McGill University, Simons Institute for the Theory of Computing and the Swiss AI Lab IDSIA. Partially supported by the ERC Starting Grant NEWNET 279352 and by Swiss National Science Foundation project 200020_144491/1.

1 Introduction

Network design is an important class of problems in Combinatorial Optimization and Theoretical Computer Science as it formulates scenarios that appear in practical settings. In particular, we might wish to design an overlay network that connects a server to clients, and this can be formulated as the *Steiner tree* problem. In a more general setting, we might have an additional constraint that the network must be able to function after link or node failures, leading to the formulation of the *survivable network design* problem. These problems are well-studied in symmetric case where a network can be represented by an undirected graph. However, in many practical settings, links in networks are not symmetric. For example, we might have different upload and download bandwidths in each connection, and sometimes, transmissions are only allowed in one direction. This motivates the study of network design problems in directed graphs, in particular, *directed Steiner* problems.

One of the most well-known directed network design problem is the *directed Steiner tree* problem (DST), which asks to find a minimum-cost subgraph that connects a given root vertex to each terminal. DST is a notorious problem as there is no known polynomial-time algorithm that gives an approximation ratio better than polynomial. A polylogarithmic approximation can be obtained only when an algorithm is allowed to run in quasi-polynomial-time [CCC$^+$99, Rot11, FKK$^+$14]. A natural generalization of DST, namely, the k *edge-connected directed Steiner tree* (k-DST) problem, where we wish to connect a root vertex to each terminal by k edge-disjoint paths, is even more mysterious as there is no known non-trivial approximation algorithm, despite having been studied and mentioned many times in literature, e.g., by Feldman et al. [FKN12], by Cheriyan et al. [CLNV14] and by Laekhanukit [Lae14].

The focus of this paper is in studying the approximability of k-DST. Let us formally describe k-DST. In k-DST, we are given a directed graph G with edge-costs $\{c_e\}_{e\in E(G)}$, a root vertex r and a set of terminals $T \subseteq V(G)$. The goal is to find a min-cost subgraph $H \subseteq G$ such that H has a k edge-disjoint directed r,t-paths from the root r to each terminal $t \in T$. Thus, removing any $k-1$ edges from H leaves at least one path from the root r to each terminal $t \in T$, and DST is the case when $k = 1$ (i.e., we need only one path). The complexity status of k-DST tends to be negative. It was shown by Cheriyan et al. [CLNV14] that the problem is at least as hard as the *label cover* problem. Specifically, k-DST admits no $2^{\log^{1-\epsilon} n}$-approximation, for any $\epsilon > 0$, unless NP $\subseteq$ DTIME($2^{\text{polylog}(n)}$). Laekhanukit [Lae14], subsequently, showed that k-DST admits no $k^{1/4-\epsilon}$-approximation unless NP = ZPP. The integrality gap of a natural LP-relaxation for k-DST is $\Omega(k/\log k)$ which holds even for a special case of *connectivity-augmentation* where we wish to increase a connectivity of a graph by one. All the lower bound results are based on the same construction which are directed acyclic graphs (DAGs) with diameter 5, i.e., any path in an input graph has length (number of edges) at most 5 (we may also say that it has 6 *layers*). Even for a very simple variant of k-DST, namely $(1, 2)$-DST, where we have two terminals, one terminal requires one path from the root and another terminal requires 2 edge-disjoint paths, it was not known whether the problem is NP-hard or polynomial-time solvable. To date, the only known positive result for k-DST is an $O(n^{kh})$-time (exact) algorithm for k-DST on DAGs [CLNV14], which thus runs in polynomial-time when kh is constant, and a folk-lore $|T|$-approximation algorithm, which can be obtained by computing min-cost k-flow for $|T|$ times, one from the root r to each terminal t and then returning the union as a solution. We emphasize that there was no known non-trivial approximation algorithm even when an input graph is "directed acyclic" and has "constant number of layers". Also, in contrast to DST, in which an $O(2^{|T|}\text{poly}(n))$-time (exact) algorithm exists for general graphs, it is not known whether k-DST for $k = 2$ and $|T| = 2$ is polynomial-time solvable if an input graph is not acyclic. This leaves challenging questions whether ones can design a non-trivial approximation algorithm for k-DST on DAGs with at most D layers, and whether ones can design a non-trivial approximation algorithm when an input graph is not acyclic.

In this paper, we make a progress toward developing a non-trivial approximation algorithm for k-DST. We present the first "non-trivial" approximation algorithm for k-DST on DAGs with D layers that achieves an approximation ratio of $O(D \cdot k^{D-1} \cdot \log n)$. Our algorithm can be extended to a special case of k-DST on "general graphs" where an instance has a D*-shallow* optimal solution, i.e., there exist k edge-disjoint r, t-paths, each of length at most D, for every terminal $t \in T$. Consequently, as our algorithm works for a general graph, we obtain an $O(D \cdot k^{D-1} \cdot \log n)$-approximation algorithm for a D-shallow instance of the k *edge-connected directed Steiner subgraph* problem, where we wish to connect every pair of terminals by k edge-disjoint paths, i.e., the set of terminal T is required to be k-edge connected in the solution subgraph (there is no root vertex in this problem).

Our algorithm is almost tight when k and D are constants because the case that $k = 1$ and $D = 3$ is essentially the *set cover* problem, which is NP-hard to approximate to within a factor of $O(\log h)$ [LY94, Fei98], and our algorithm achieves the same approximation ratio. We also remark that the $k^{1/4-\epsilon}$-hardness instance of k-DST is a DAG with 6 layers, and our algorithm gives $O(k^5 \log n)$-approximation for this special case. For $k = 1$, we obtain a slightly better bound of $O(D \log h)$, thus giving an LP-based $O(\log^3 h)$-approximation algorithm for DST as a by product.

The key idea of our algorithm is to formulate an LP-relaxation with a special property that a fractional solution can be embedded into a tree instance of the *group Steiner tree* problem (GST). Thus, we can apply the GKR Rounding algorithm in [GKR00] for GST on trees to round the fractional solution. However, embedding of an LP-solution to a tree instance of GST does not preserve connectivity. Also, it does not lead to a reduction from k-DST to the k edge-connected variant of GST, namely, k-GST. Hence, our algorithm is, although simple, not straightforward.

1.1 Our Results

Our main result is an $O(D \cdot k^{D-1} \cdot \log n)$-approximation algorithm for k-DST on a D-shallow instance, which includes a special case that an input graph is directed acyclic and has at most D layers.

Theorem 1. *Consider the k edge-connected directed Steiner tree problem. Suppose an input instance has an optimal solution H^* in which, for every terminal $t \in T$, H^* has k edge-disjoint r, t-paths such that each path has length at most D. Then there exists an $O(D \cdot k^{D-1} \cdot \log n)$-approximation algorithm. In particular, there is an $O(D \cdot k^{D-1} \cdot \log n)$-approximation algorithm for k-DST on a directed acyclic graph with D layers.*

For the case $k = 1$, our algorithm yields a slightly better guarantee of $O(D \log h)$. Thus, we have as by product an LP-based approximation algorithm for DST. Applying Zelikovsky's height-reduction theorem [Zel97, HRZ01], this implies an LP-based quasi-polynomial-time $O(\log^3 h)$-approximation algorithm for DST. (The algorithm runs in time $O(\text{poly}(n^D)$ and has approximation ratio $O(h^{1/D} \cdot D^2 \log h)$.)

Theorem 2 also implies an algorithm of the same (asymptotic) approximation ratio for a D-shallow instance of the k edge-connected directed Steiner subgraph problem, where we wish to find a subgraph H such that the set of terminal T is k-edge-connected in H. To see this, we invoke the algorithm in Theorem 2 as follows. Take any terminal $t^* \in T$ as a root vertex of a k-DST instance. Then we apply the algorithm for k-DST to find a subgraph H^{out} such that every terminal is k edge-connected from t^*. We apply the algorithm again to find a subgraph H^{in} such that every terminal is k edge-connected to t^*. Then the set of terminal T is k-edge connected in the graph $H^{out} \cup H^{in}$ by transitivity of edge-connectivity. The cost incurred by this algorithm is at most twice that of the algorithm in Theorem 2. Thus, we have the following theorem as a corollary of Theorem 2

Theorem 2. *Consider the k edge-connected directed Steiner subgraph problem. Suppose an input instance has an optimal solution H^* in which, for every pair of terminals $s, t \in T$, H^* has k edge-disjoint s, t-paths such that each path has length at most D. Then there exists an $O(D \cdot k^{D-1} \cdot \log n)$-approximation algorithm.*

Overview of our algorithm The key idea of our algorithm is to embed an LP solution for k-DST to a standard LP of GST on a tree. (We emphasize that we embed the LP solution of k-DST to that of GST not k-GST.) At first glance, a reduction from k-DST to GST on trees is unlikely to exist because any such reduction would destroy all the connectivity information. We show, however, that such tree-embedding exists, but we have to sacrifice running-time and cost to obtain such embedding.

The reduction is indeed the same as a folk-lore reduction from DST to GST on trees. That is, we list all rooted-paths (paths that start from the root vertex) of length at most D in an input graph and form a suffix tree. In the case of DST, if there is an optimal solution which is a tree of height D, then it gives an approximation preserving reduction from GST to DST which blows up the size (and thus the running time) of the instance to $O(n^D)$. Unfortunately, for the case of k-DST with $k > 1$, this reduction does not give an equivalent reduction from k-DST to k-GST on trees. The reduction is valid in one direction, i.e., any feasible solution to k-DST has a corresponding feasible solution to the tree-instance of k-GST. However, the converse is not true as a feasible solution to the tree-instance of k-GST might not give a feasible solution to k-DST. Thus, our reduction is indeed an "invalid" reduction from k-DST to a tree instance of "GST" (the case $k = 1$).

To circumvent this problem, we formulate an LP that provides a connection between an LP solution on an input k-DST instance and an LP solution of a tree-instance of GST. Thus, we can embed an LP solution to an LP-solution of GST on a (very large) tree. We then round the LP solution using the GKR Rounding algorithm for GST on trees [GKR00]. This algorithm, again, does not give a feasible solution to k-DST as each integral solution we obtain only has "connectivity one" and thus is only feasible to DST. We cope with this issue by using a technique developed by Chalermsook et al. in [CGL15]. Specifically, we sample a sufficiently large number of DST solutions and show that the union of all these solutions is feasible to k-DST using cut-arguments.

Each step of our algorithm and the proofs are mostly standard, but ones need to be careful in combining each step. Otherwise, the resulting graph would not be feasible to k-DST.

Organization. We provide definitions and notations in Section 2. We start our discussion by presenting a reduction from DST to GST in Section 3. Then we discuss properties of minimal solutions for k-DST in Section 4. We present standard LPs for k-DST and GST in Section 5 and formulate a stronger LP-relaxation for k-DST in Section 6. Then we proceed to present our algorithm in Section 7. Finally, we provide some discussions in Section 8.

2 Preliminaries

We use standard graph terminologies. We refer to a directed edge (u, v), shortly, by uv (i.e., u and v are head and tail of uv, respectively), and we refer to an undirected edge by $\{u, v\}$. For a (directed or undirected) graph G, we denote by $V(G)$ and $E(G)$ the sets of vertices and edges of G, respectively. If a graph G is associated with edge-costs $\{c_e\}_{e\in E(G)}$, then we denote the cost of any subgraph $H \subseteq G$ by $\mathsf{cost}(H) = \sum_{e\in E(H)} c_e$. For any path P, we use *length* to mean the number of edges in a path P and use *cost* to mean the total costs of edges in P.

In the *directed Steiner tree* problem (DST), we are given a directed graph G with edge-costs $\{c_e\}_{e\in E(G)}$, a root vertex r and a set of terminals $T \subseteq V(G)$. The goal is to find a min-cost subgraph $H \subseteq G$ such that H has a directed path from the root r to each terminal $t \in T$. A generalization of DST is the k *edge-connected directed Steiner tree* problem (k-DST). In k-DST, we are given the same input as in DST plus an integer k. The goal is to find a min-cost subgraph H that has k edge-disjoint paths from the root r to each terminal $t \in T$. The k *edge-connected directed Steiner subgraph* problem is a variant of k-DST, where there is no root vertex, and the goal is to find a min-cost subgraph H such that the set of terminals T is k edge-connected in H.

The problems on undirected graphs that are closely related to of DST and k-DST are the *group Steiner tree* problem (GST) and the k *edge-connected group Steiner tree* problem (k-GST). In GST, we are given an undirected graph $\mathcal{G}$ with edge-costs $\{c_e\}_{e\in E(\mathcal{G})}$, a root vertex r and a collection of subset of vertices $\{\mathcal{T}_i\}_{i=1}^{h}$ called groups. The goal is to find a a min-cost subgraph $\mathcal{H}$ that connects r to each group $\mathcal{T}_i$. In k-GST, the input consists of an additional integer k, and the goal is to find a min-cost subgraph H with k edge-disjoint $r, \mathcal{T}_i$-paths for every group T_i.

Consider an instance of DST (resp., k-DST). We denote by Q the set of all paths in G that start from the root r. The set of paths in Q that end with a particular pattern, say $\sigma = (v_1, \ldots, v_q)$, is denoted by $Q(\sigma)$. This pattern σ can be a vertex v, an edge e or a path $\sigma = (v_1, \ldots, v_q)$ in G. For example, $Q(u, v, w)$ consists of paths P of the form $P = (r, \ldots, u, v, w)$. We say that a path P ends at a vertex v (resp., an edge e) if v (resp., e) is the last vertex (resp., edge) of P.

We may consider only paths with particular length, say D. We denote by Q_D the set of paths that start at r and has length at most D. The notation for Q_D is analogous to Q, e.g., $Q_D(uv) \subseteq Q_D$ is the set of paths in Q_D that end at an edge uv. A concatenation of a path p with an edge e or a vertex v are denoted by $p + e$ and $p + v$, respectively. For example, $(u_1, \ldots, u_\ell) + vw = (u_1, \ldots, u_\ell, v, w)$.

Given a subset of vertices S, the set of edges entering S is denoted by

$$\delta^-(S) = \{uv \in E : u \in S, v \notin S\}$$

The indegree of S is denoted by $\mathsf{indeg}(S) = |\delta^-(S)|$. Analogously, we use $\delta^+(S)$ and $\mathsf{outdeg}(S)$ for the set of edges leaving S. For undirected graphs, we simply use the notations $\delta(S)$ and $\mathsf{deg}(S)$.

We say that a feasible solution H to k-DST is D-*shallow* if, for every terminal $t \in T$, there exists a set of k edge-disjoint r, t-paths in H such that every path has length at most D. An instance of k-DST that has an optimal D-shallow solution is called a D-shallow instance. We also use the term D-shallow analogously for k-GST and the k edge-connected Steiner subgraph problem.

To distinguish between the input of k-DST (which is a directed graph) and k-GST (which is an undirected graph), we use script fonts, e.g., $\mathcal{G}$, to denote the input of k-GST. Also, we use $\mathcal{Q}$ to denote the set of all paths from the root r to any vertex v in the graph $\mathcal{G}$. The cost of a set of edges F (or a graph) is defined by a function $\mathsf{cost}(F) = \sum_{e\in F} c_e$. At each point, we consider only one instance of k-DST (respectively, k-GST). So, we denote the cost of the optimal solution to k-DST by opt_{kDST} (respectively, opt_{kGST}).

3 Reduction from Directed Steiner Tree to Group Steiner Tree

In this section, we describe a reduction $\mathcal{R}$ from DST to GST. We recall that Q denotes all the r,v-paths in a DST instance G. The reduction is by simply listing paths in the directed graph G as vertices in a tree $\mathcal{G} = \mathcal{R}(G)$ and joining each path p to $p + e$ if $p + e$ is a path in G. In fact, $\mathcal{R}(G)$ is a *suffix tree* of paths in Q. To be precise,

$$V(\mathcal{G}) = \{p : p \text{ is an } r,v\text{-path in } G\}$$
$$E(\mathcal{G}) = \{\{p, p+e\} : \text{both } p \text{ and } p+e \text{ are paths in } G \text{ starting at } r\}$$

We set the cost of edges of $\mathcal{G}$ to be $c_{\{p,p+e\}} = c_e$. Since the root r has no incoming edges in G, r maps to a unique vertex $(r) \in \mathcal{G}$, and we define (r) as the root vertex of the GST instance. We will abuse r to mean both the root of DST and its corresponding vertex of GST. For each terminal $t_i \in T$, define a group of the GST instance as

$$\mathcal{T}_i := Q(t_i) = \{p \subseteq G : p \text{ is an } r,t_i\text{-path in } G\}$$

It is easy to see that the reduction $\mathcal{R}$ produces a tree, and there is a one-to-one mapping between a path in the tree $\mathcal{G} = \mathcal{R}(G)$ and a path in the original graph G. Thus, any tree in G corresponds to a subtree of $\mathcal{R}(G)$ (but not vice versa), which implies that the reduction $\mathcal{R}$ is approximation-preserving (i.e., $\mathsf{opt}_{DST} = \mathsf{opt}_{GST}$). Note, however, that the size of the instance blows up from $O(n+m)$ to $O(n^D)$, where D is the length of the longest path in G. The reduction holds for general graphs, but it is approximation-preserving only if the DST instance is D-shallow, i.e., it has an optimal solution H^* such that any r,t_i-path in H^* has length at most D, for all terminals $t_i \in T$. However, Zelikovsky [Zel97, HRZ01] showed that the *metric completion* of G always contains a D-shallow solution with cost at most $D|V(G)|^{1/D}$ of an optimal solution to DST. (This is now known as Zelikovsky's height reduction theorem.) Thus, we may list only paths of length at most D from the metric completion. We denote the reduction that lists only paths of length at most D by $\mathcal{R}_D$.

4 Properties of Minimal Solutions to k-DST

In this section, we provide structural lemmas which are building blocks in formulating a strong LP-relaxation for k-DST. These lemmas characterize properties of a minimal solution to k-DST.

Lemma 3. *Let H be any minimal solution to k-DST. Then H has at most k edge-disjoint r,v-paths, for any vertex $v \in V(H)$.*

Proof. Suppose to a contrary that H has $k+1$ edge-disjoint r,v-paths, for some vertex $v \in V(H)$. Then v must have indegree at least $k+1$ in H. We take one of the $k-1$ edges entering v, namely, uv. By minimality of H, removing uv results in a graph $H' = H - uv$ that has less than k edge-disjoint r,t_i-paths for some terminal $t_i \in T$. Thus, by Menger's theorem, there must be a subset of vertices $S \subseteq V$ such that $t_i \in S$, $r \in V - S$ and $\mathsf{indeg}_{H'}(S) \leq k-1$. Observe that we must have

uv in $\delta^-_H(S)$ because H is a feasible solution to k-DST, which means that $v \in S$. Since we remove only one edge uv from H, the graph H' must have k edge-disjoint r, v-paths. But, this implies that $\mathsf{indeg}_{H'}(S) \geq k$, a contradiction. □

Lemma 4. *Let H be any minimal solution to k-DST. Any vertex $v \in V(H)$ has indegree exactly $\lambda(v)$, where $\lambda(v)$ is the maximum number of edge-disjoint r, v-paths in H.*

Proof. The proof follows a standard uncrossing argument. Assume a contradiction that v has indegree at least $\lambda(v) + 1$ in H. By Menger's theorem, there is a subset of vertices $U \subseteq V$ such that $\mathsf{indeg}_H(U) = \lambda(v)$, $v \in U$ and $r \notin U$ that separates v from r. We assume that U is a minimum such set. Since $\mathsf{indeg}_H(v) > \lambda(v)$, there is an edge $uv \in E(H)$ that is not contained in $\delta^-_H(U)$, i.e., $u, v \in U$.

By minimality of H, removing uv results in the graph $H' = H - uv$ such that H' has less than k edge-disjoint r, t_i-path for some terminal $t_i \in T$. Thus, by Menger's theorem, there is a subset of vertices W such that $t_i \in W$, $r \notin W$, $uv \in \delta^-_H(W)$ and $\mathsf{indeg}_H(W) = k$. (The latter is because H is a feasible solution to k-DST.)

Now we apply an uncrossing argument to U and W. By submodularity of indeg_H, we have

$$\mathsf{indeg}_H(U) + \mathsf{indeg}_H(W) \geq \mathsf{deg}_H(U \cap W) + \mathsf{deg}_H(U \cup W)$$

Observe that $v \in U \cap W$, $t \in U \cup W$ and $r \notin S \cup S'$. So, by the edge-connectivity of v and t,

$$\mathsf{indeg}_H(U \cap W) \geq \lambda(v) \quad \text{and} \quad \mathsf{indeg}_H(U \cup W) \geq k \tag{1}$$

The sum of the left-hand side of Eq (1) is

$$\mathsf{indeg}_H(U) + \mathsf{indeg}_H(W) = k + \lambda(v).$$

So, we conclude that

$$\mathsf{indeg}_H(U \cap W) = \lambda(v) \quad \text{and} \quad \mathsf{indeg}_H(U \cup W) = k$$

Consequently, we have the set $U' = U \cap W$ such that $\mathsf{indeg}_H(U') = \lambda(v)$, $v \in U'$ and $r \notin U'$ that separates v from r. Since $u \notin W$, we know that U' is strictly smaller than U. This contradicts to the minimality of U. □

The following is a corollary of Lemma 3 and Lemma 4

Corollary 5. *Let H be a minimal solution to k-DST. Then any vertex $v \in V(H)$ has indegree at most k.*

The next lemma follows from Corollary 5.

Lemma 6. *Consider any minimal solution H to k-DST (which is a simple graph). For any edge $e \in E(H)$ and $\ell \geq 2$, there are at most $k^{\ell-2}$ paths in H with length at most ℓ that start at the root r and ends at e. That is, $|Q^H_\ell(e)| \leq k^{\ell-2}$ for all $e \in E(H)$, where $Q^H_\ell(e)$ is the set of r, v-paths of length ℓ in H.*

Proof. We prove by induction. The base case $\ell = 2$ is trivial because any rooted path of length at most 2 cannot have a common edge.

Assume, inductively, that $|Q^H_{\ell-1}(e)| \leq k^{\ell-3}$ for some $\ell \geq 3$. Consider any edge $vw \in E(H)$. By Corollary 5, v has indegree at most k. Thus, there are at most k edges entering v, namely, $u_1v, \ldots, u_dv$, where $d = \mathsf{indeg}(v)$. By the induction hypothesis, each edge is the last edge of at most $k^{\ell-3}$ paths in $Q^H_{\ell-1}$. Thus, we have at most $d \cdot k^{\ell-3} \leq k^{\ell-2}$ paths that end at uv. That is,

$$|Q^H_\ell(vw)| \leq \sum_{i=1}^{d} |Q^H_{\ell-1}(u_dv)| \leq \sum_{i=1}^{d} k^{\ell-3} = d \cdot k^{\ell-3} \leq k^{\ell-2}.$$

$\square$

5 Standard LPs for k-DST and GST

In this section, we describe standard LPs for k-DST and GST. Each LP consists of two sets of variables, a variable x_e on each edge e and a variable f^i_p on each path p and a terminal t_i. The variable x_e indicates whether we choose an edge e in a solution. The variable f^i_p is a flow-variable on each path and thus can be written in a compact form using a standard flow formulation.

$$\text{LP-k-DST}\left\{\begin{array}{lrll} \min & \sum_{e\in E(G)} c_e x_e & & \\ \text{s.t.} & \sum_{p\in Q(t_i):e\in E(p)} f^i_p & \leq x_e & \forall e\in E(G), \forall t_i \in T \\ & \sum_{p\in Q(t_i)} f^i_p & \geq k & \forall t_i \in T \\ & x_e & \leq 1 & \forall e \in E(G) \\ & x_e & \geq 0 & \forall e \in E(G) \\ & f^i_p & \geq 0 & \forall p \in Q(t_i), \forall t_i \in T. \end{array}\right.$$

The standard LP for GST is similar to LP-k-DST.

$$\text{LP-GST}\left\{\begin{array}{lrll} \min & \sum_{e\in E(\mathcal{G})} c_e x_e & & \\ \text{s.t.} & \sum_{v\in\mathcal{T}_i}\sum_{p\in\mathcal{Q}(v):e\in E(p)} f^i_p & \leq x_e & \forall e\in E(\mathcal{G}), \forall i = 1,2,\ldots,h \\ & \sum_{v\in\mathcal{T}_i}\sum_{p\in\mathcal{Q}(v)} f^i_p & \geq 1 & \forall i = 1,2,\ldots,h \\ & x_e & \leq 1 & \forall e \in E(\mathcal{G}) \\ & x_e & \geq 0 & \forall e \in E(\mathcal{G}) \\ & f^i_p & \geq 0 & \forall p \in \mathcal{Q}, \forall i = 1,2,\ldots,h \end{array}\right.$$

6 A Strong LP-relaxation for for k-DST

In this section, we formulate a strong LP-relaxation for k-DST that allows us to embed a fractional solution into an LP solution of LP-GST on a tree.

$$
\text{LP-k-DST*}\begin{cases}
\min & \sum_{e\in E} c_e x_e & & & \\
\text{s.t.} & \sum_{p\in Q(t_i):e\in E(p)} f^i_p & \le x_e & \forall e\in E(G), \forall t_i\in T & \\
& \sum_{p\in Q(t_i)} f^i_p & \ge k & \forall t_i \in T & \\
& \sum_{p\in Q(t_i):q\subseteq p} f^i_p & \le y_q & \forall q\in Q, \forall t_i\in T & \text{(Subflow Capacity)}\\
& \sum_{p\in Q_\ell(e)} y_p & \le \max\{1, k^{\ell-2}\}\cdot x_e & \forall e\in E, \forall \ell\ge 1 & \text{(Aggregating } k\text{-Flow)}\\
& x_e & \le 1 & \forall e\in E(G) & \\
& x_e & \ge 0 & \forall e\in E(G) & \\
& f^i_p & \ge 0 & \forall p\in Q(t_i), \forall t_i\in T & \\
& y_p & \ge 0 & \forall p\in Q &
\end{cases}
$$

For D-shallow instances of k-DST, we replace Q by Q_D to restrict length of paths to be at most D. The next lemma shows that LP-k-DST* is an LP-relaxation for k-DST.

Lemma 7. *LP-k-DST* is an LP-relaxation for k-DST. Moreover, replacing Q by Q_D gives an LP-relaxation for k-DST on D-shallow instances.*

Proof. LP-k-DST* is, in fact, obtained from LP-k-DST (which is a standard LP) by adding a new variable y_p and two constraints.

(1) **Subflow-Capacity:** $\displaystyle\sum_{p\in Q(t_i):q\subseteq p} f^i_p \le y_q, \forall q\in Q, \forall t_i\in T.$

(2) **Aggregating k-Flow:** $\displaystyle\sum_{p\in Q_\ell(e)} y_p \le \max\{1, k^{\ell-2}\}\cdot x_e, \forall e\in E, \forall \ell\ge 1.$

To show that these two constraints are valid for k-DST, we take a minimal feasible (D-shallow) solution H of k-DST. We define a solution (x, f, y) to LP-k-DST as below.

$$
x_e = \begin{cases} 1 & \text{if } e\in E(H)\\ 0 & \text{otherwise}\end{cases}
\qquad
y_p = \begin{cases} 1 & \text{if } p\subseteq H \wedge p\in Q\\ 0 & \text{otherwise}\end{cases}
$$
$$
f^i_p = \begin{cases} 1 & \text{if } p\subseteq H \wedge p\in Q(t_i)\\ 0 & \text{otherwise}\end{cases}
$$

By construction, $f^i_p = 1$ implies that $y_p = 1$. Thus, (x, f, y) satisfies the Subflow-Capacity constraint. By minimality of H, Corollary 6 implies that even if we list all the paths of length $\ell \ge 2$ in H, at most $k^{\ell-2}$ of them end at the same edge, and we know that rooted paths of length one share no edge (given that H is a simple graph). Thus, (x, f, y) satisfies the Aggregating k-Flow constraint. Consequently, these two constraints are valid for k-DST.

On the other hand, any integral solution that is not feasible to k-DST could not satisfy the constraints of LP-k-DST* simply because LP-k-DST* contains the constraints of LP-k-DST, which is a standard LP for k-DST. Thus, LP-k-DST* is an LP-relaxation for k-DST.

The proof for the case of D-shallow instances is the same as above except that we take H as a minimal D-shallow solution and replace Q by Q_D. □

7 An Approximation Algorithm for k-DST

In this section, we present an approximation algorithm for k-DST on a D-shallow instance. Our algorithm is simple. We solve LP-k-DST* on an input graph G and then embed an optimal fractional solution (x, f, y) to an LP-solution $(\hat{x}, \hat{f})$ of LP-GST on the tree $\mathcal{R}(G)$. We lose a factor of $O(k^{D-2})$ in this process. As we now have a tree-embedding of an LP-solution, we can invoke the GKR Rounding algorithm [GKR00] to round an LP-solution on the tree $\mathcal{R}(G)$. Our embedding guarantees that any edge-set of size $k-1$ in the original graph G never maps to an edge-set in the tree $\mathcal{G} = \mathcal{R}(G)$ that separates r and $\mathcal{T}_i = Q(t_i)$ in $\mathcal{G}$. So, the rounding algorithm still outputs a feasible solution to GST with constant probability even if we remove edges in the tree $\mathcal{G}$ that correspond to a subset of $k-1$ edges in G. Consequently, we only need to run the algorithm for $O(D \cdot k \cdot \log n)$ times to boost the probability of success so that, for any subset of $k-1$ edges and any terminal $t_i \in T$, we have at least one solution that contains an r, t_i-path using none of these $k-1$ edges. In other words, the union of all the solutions satisfies the connectivity requirement. Our algorithm is described in Algorithm 1.

Algorithm 1 Algorithm for k-DST

```
Solve LP-k-DST* and obtain an optimal solution $(x, f, y)$.
Map $(x, f, y)$ to a solution $(\hat{x}, \hat{f})$ to LP-GST on $\mathcal{G} = \mathcal{R}(G)$.
for $i = 1$ to $2Dk \log_2 n$ do
  Run GKR Rounding on $(\hat{x}, \hat{f})$ to get a solution $\mathcal{Z}_i$.
  Map $\mathcal{Z}_i$ back to a subgraph $Z_i$ of $G$.
end for
return $H = \bigcup_i Z_i$ as a solution to k-DST.
```

We map a solution (x, f, y) of LP-k-DST* on G to a solution $(\hat{x}, \hat{f})$ of LP-GST on the tree $\mathcal{G} = \mathcal{R}(G)$ as below. Note that there is a one-to-one mapping between a path in G and a path in the tree $\mathcal{G}$.

$$\begin{aligned} \hat{x}_{\{p,p+e\}} &:= y_{p+e} && \text{for all } p+e \in Q \\ \hat{f}^i_p &:= f^i_p && \text{for all } p \in Q \text{ and for all } t_i \in T \end{aligned}$$

7.1 Cost Analysis

We show that $\mathsf{cost}(\hat{x}, \hat{f}) \leq k^{D-2} \cdot \mathsf{cost}(x, f, y)$.

Lemma 8. *Consider a solution $(\hat{x}, \hat{f})$ to LP-GST, which is mapped from a solution (x, f) of LP-k-DST* when an input k-DST instance is D-shallow, for $D \geq 2$. The cost of $(\hat{x}, \hat{f})$ is at most $\mathsf{cost}(\hat{x}, \hat{f}) \leq k^{D-2} \cdot \mathsf{cost}(x, f)$.*

Proof. By the constraint $\sum_{p \in Q_\ell(e)} f_p \leq \max\{1, k^{\ell-2}\} \cdot x_e$, we have that

$$\begin{aligned} \mathsf{cost}(\hat{x}, \hat{f}) &= \textstyle\sum_{e' \in E(\mathcal{G})} c_{e'} \hat{x}_{e'} && = \textstyle\sum_{e \in E(G)} \sum_{\{p,p+e\} \in E(\mathcal{G})} c_e \hat{x}_{\{p,p+e\}} \\ &= \textstyle\sum_{e \in E(G)} \sum_{p+e \in Q} c_e f_{p+e} && = \textstyle\sum_{e \in E(G)} \sum_{p \in Q(e)} c_e f_p \\ &= \textstyle\sum_{e \in E(G)} \left(c_e \cdot \sum_{p \in Q(e)} f_p \right) && \leq \textstyle\sum_{e \in E(G)} c_e \cdot k^{D-2} \cdot x_e \\ &= k^{D-2} \cdot \mathsf{cost}(x, f). \end{aligned}$$

□

It can be seen from Algorithm 1 and Lemma 8 that the algorithm outputs a solution H with cost at most $O(Dk^{D-1}\log n)\cdot\mathsf{cost}(x,f)$. Thus, H is an $O(Dk^{D-1}\log n)$-approximate solution. It remains to show that H is feasible to k-DST.

7.2 Feasibility Analysis

Now we show that H is feasible to k-DST with at least constant probability. To be formal, consider any subset $F\subseteq E(G)$ of $k-1$ edges. We map F to their corresponding edges $\mathcal{F}$ in the tree $\mathcal{G}$. Thus, $\mathcal{F}:=\{\{P,P+e\}:P+e\in Q\wedge e\in F\}$.

Observe that no vertices in $\mathcal{G}-\mathcal{F}$ correspond to paths that contain an edge in F. Thus, we can define an LP solution (y^F,z^F) for LP-GST on the graph $\mathcal{G}-\mathcal{F}$ as follows.

$$y^F_e \quad=\begin{cases}\hat{x}_e & \text{if } e\notin\mathcal{F}\\ 0 & \text{otherwise}\end{cases}\qquad\qquad z^{F,i}_p \quad=\begin{cases}\hat{f}^i_p & \text{if } E(p)\cap\mathcal{F}=\emptyset\\ 0 & \text{otherwise}\end{cases}$$

We show that (y^F,z^F) is feasible to LP-GST on $\mathcal{G}-\mathcal{F}$.

Lemma 9. *For any subset of edges $F\subseteq E(G)$, define (y^F,z^F) from $(\hat{x},\hat{f})$ as above. Then (y^F,z^F) is feasible to LP-GST on $\mathcal{G}-\mathcal{F}$.*

Proof. First, observe that $z^{F,i}_p>0$ only if a path p contains no edges in $\mathcal{F}$. So, by construction, (y^F,z^F) satisfies $z^{F,i}_p=\hat{f}^i_p\leq\hat{x}_e=y^F_e$ for all $e\in E(p)$. Hence, (y^F,z^F) satisfies the capacity constraint.

Next we show that (y^F,z^F) satisfies the connectivity constraint. Consider the solution (x,f,y) to LP-k-DST*. By the feasibility of (x,f,y) and the Max-Flow-Min-Cut theorem, the graph $G-F$ with capacities $\{x_e\}_{e\in G-F}$ can support a flow of value one from r to any terminal t_i. This implies that $\sum_{p\in Q(t_i):E(p)\cap F=\emptyset}f^i_p\geq 1$. Consequently, we have

$$\begin{aligned}\textstyle\sum_{p\in Q(t_i):E(p)\cap F=\emptyset}f^i_p &= \textstyle\sum_{p\in\mathcal{T}_i:E(p)\cap F=\emptyset}\sum_{p'\in\mathcal{Q}(v)}\hat{f}^i_{p'}\\ &= \textstyle\sum_{v\in\mathcal{T}_i}\sum_{p'\in\mathcal{Q}(v):E(p')\cap\mathcal{F}=\emptyset}\hat{f}^i_{p'}\\ &= \textstyle\sum_{v\in\mathcal{T}_i}\sum_{p'\in\mathcal{Q}(v):E(p')\cap\mathcal{F}=\emptyset}z^{F,i}_{p'}\\ &= \textstyle\sum_{v\in\mathcal{T}_i}\sum_{p'\in\mathcal{Q}(v)}z^{F,i}_{p'}\\ &\geq 1.\end{aligned}$$

All the other constraints are satisfied because (y^F,z^F) is constructed from $(\hat{x},\hat{f})$. Thus, (y^F,z^F) is feasible to LP-GST on $\mathcal{G}-\mathcal{F}$. □

Lemma 9 implies that we can run the GKR Rounding algorithm on (y^F,z^F). The following is the property of GKR Rounding.

Lemma 10 ([GKR00]). *There exists a randomized algorithm such that, given a solution $(\hat{x},\hat{f})$ to LP-GST on a tree $\mathcal{G}$ with height D, the algorithm outputs a subgraph $\mathcal{H}\subseteq\mathcal{G}$ so that the probability that any subset of vertices $U\subseteq V(\mathcal{G})$ is connected to the root is at least*

$$\Pr[\mathcal{H}\text{ has an }r,U\text{-path.}]\geq\frac{\sum_{v\in U}\sum_{p\in\mathcal{Q}(v)}\hat{f}^i_p}{O(D)}$$

Moreover, the probability that each edge is chosen is at most $\hat{x}_e$. That is, $\mathbb{E}[\mathsf{cost}(\mathcal{H})]=\mathsf{cost}(\hat{x},\hat{f})$. The running time of the algorithm is $O(|E(\mathcal{G})|+|V(\mathcal{G})|)$.

Since $(y^F, z^F) \leq (\hat{x}, \hat{f})$ (coordinate-wise), we can show that running GKR Rounding on $(\hat{x}, \hat{f})$ simulates the runs on (y^F, z^F) for all $F \subseteq E(G)$ with $|F| \leq k-1$, simultaneously.

Lemma 11. *Let $\mathcal{H}$ be a subgraph of $\mathcal{G}$ obtained by running GKR Rounding on $(\hat{x}, \hat{f})$, and let H be a subgraph of G corresponding to $\mathcal{H}$. Then, for any subset of edges $F \subseteq E(G)$ with $|F| \leq k-1$ and for any terminal $t_i \in T$,*

$$\Pr[H - F \textit{ has an } r, t_i\textit{-path}] \geq \frac{1}{O(D)}.$$

Proof. Let us briefly describe the work of GKR Rounding. The algorithm marks each edge e in the tree with probability $x_e / x_{\varrho(e)}$, where $\varrho(e)$ is the parent of an edge e in $\mathcal{G}$, which is unique. Then the algorithm picks an edge e if all of its ancestors are marked. Clearly, removing any set of edges $\mathcal{F}$ from $\mathcal{G}$ only affects paths that contain an edge in $\mathcal{F}$.

Let (y^F, z^F) be defined from $(\hat{x}, \hat{f})$ as above. This LP solution is defined on a graph $\mathcal{G} - \mathcal{F}$. Thus, the probability of success is not affected by removing $\mathcal{F}$ from the graph. By Lemma 9, we can run GKR Rounding on (y^F, z^F) and obtain a subgraph $\mathcal{H}^F$ of $\mathcal{G} - \mathcal{F}$. Since $z_p^F \leq \hat{f}_p$ for all paths $p \in \mathcal{Q}$ and $z_p^F = 0$ for all $p \in \mathcal{Q} : E(p) \cap \mathcal{F} \neq \emptyset$, we have from Lemma 10 and Lemma 9 that

$$\begin{aligned}
\Pr[H - F \text{ has an } r, t_i\text{-path}] &= \Pr[\mathcal{H} - \mathcal{F} \text{ has an } r, \mathcal{T}_i\text{-path}] \\
&\geq \Pr[\mathcal{H}^F \text{ has an } r, \mathcal{T}_i\text{-path}] \\
&\geq \frac{\sum_{v \in \mathcal{T}_i} \sum_{p \in \mathcal{Q}(v)} z_p^F}{O(D)} \\
&\geq \frac{1}{O(D)}.
\end{aligned}$$

□

Finally, we recall that Algorithm 1 employs GKR Rounding on $(\hat{x}, \hat{f})$ for $2Dk \log_2 n$ times. So, for any subset of $k-1$ edges $F \subseteq E(G)$ and for any terminal $t_i \in T$, there exists one subgraph that has an r, t_i-path that contains no edge in F with large probability. In particular, the union is a feasible solution to k-DST with at least constant probability.

Lemma 12. *Consider the run of Algorithm 1. The solution subgraph $H = \bigcup_i Z_i$ is a feasible solution to k-DST with probability at least $1/n$.*

Proof. For $i = 1, 2, \ldots, 2Dk \log_2 n$, let Z_i be a subgraph of G obtained by running GKR Rounding on $(\hat{x}, \hat{f})$ and mapping the solution back to a subgraph of G as in Algorithm 1. By Lemma 11, $Z_i - F$ has an r, t_i-path with probability $\Omega(1/D)$. Since each Z_i is sampled independently, we have

$$\Pr[\forall i\, Z_i - F \text{ has no } r, t_i\text{-path}] \leq \left(1 - \frac{1}{O(D)}\right)^{2Dk \log_2 n} \leq \left(\frac{1}{e}\right)^{2k \log_2 n} \leq n^{-2k}.$$

We have at most $|E(G)|^{k-1} \leq n^{2(k-1)}$ such sets F and at most $|T| \leq n$ terminals. So, there are at most n^{2k-1} bad events where there exists an edge-set of size $k-1$ that separates the root r and some terminal $t_i \in T$. Therefore, by union bound, $H = \bigcup_i Z_i$ is a feasible solution to k-DST with probability at least $1/n$. □

This completes the proof of Theorem 2. Note that, for the case of DST ($k = 1$), we only need to run GKR Rounding for $O(D \log h)$ times, thus implying an approximation ratio of $O(D \log h)$.

8 Conclusion and Discussion

We presented the first non-trivial approximation algorithm for k-DST in a special case of a D-shallow instance, which exploits the reduction from DST to GST. We hope that our techniques will shed some light in designing an approximation algorithm for k-DST in general case and perhaps lead to a bi-criteria approximation algorithm in the same manner as in [CGL15].

One obstruction in designing an approximation algorithm in directed graphs is that there is no "true" (perhaps, probabilistic) tree-embedding for directed graphs. Both devising a tree-embedding for directed graphs and designing an approximation algorithm for k-DST with $k \geq 2$ are big open problems in the area. Another open problem, which is considered as the most challenging one by many experts, is whether there exists a polynomial-time algorithm for DST that yields a sub-polynomial approximation ratio.

Acknowledgements. Our work was inspired by the works of Rothvoß [Rot11] and Friggstad et al. [FKK+14] and by discussions with Joseph Cheriyan and Lap Chi Lau. We also thank Zachary Friggstad for useful discussions.

References

[CCC+99] Moses Charikar, Chandra Chekuri, To-Yat Cheung, Zuo Dai, Ashish Goel, Sudipto Guha, and Ming Li. Approximation algorithms for directed steiner problems. *J. Algorithms*, 33(1):73–91, 1999. 2

[CGL15] Parinya Chalermsook, Fabrizio Grandoni, and Bundit Laekhanukit. On survivable set connectivity. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 25–36, 2015. 4, 13

[CLNV14] Joseph Cheriyan, Bundit Laekhanukit, Guyslain Naves, and Adrian Vetta. Approximating rooted steiner networks. *ACM Transactions on Algorithms*, 11(2):8:1–8:22, 2014. 2

[Fei98] Uriel Feige. A threshold of ln n for approximating set cover. *J. ACM*, 45(4):634–652, 1998. 3

[FKK+14] Zachary Friggstad, Jochen Könemann, Young Kun-Ko, Anand Louis, Mohammad Shadravan, and Madhur Tulsiani. Linear programming hierarchies suffice for directed steiner tree. In *Integer Programming and Combinatorial Optimization - 17th International Conference, IPCO 2014, Bonn, Germany, June 23-25, 2014. Proceedings*, pages 285–296, 2014. 2, 13

[FKN12] Moran Feldman, Guy Kortsarz, and Zeev Nutov. Improved approximation algorithms for directed steiner forest. *J. Comput. Syst. Sci.*, 78(1):279–292, 2012. 2

[GKR00] Naveen Garg, Goran Konjevod, and R. Ravi. A polylogarithmic approximation algorithm for the group steiner tree problem. *J. Algorithms*, 37(1):66–84, 2000. 3, 4, 10, 11

[HRZ01] Christopher S. Helvig, Gabriel Robins, and Alexander Zelikovsky. An improved approximation scheme for the group steiner problem. *Networks*, 37(1):8–20, 2001. 3, 6

[Lae14] Bundit Laekhanukit. Parameters of two-prover-one-round game and the hardness of connectivity problems. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1626–1643, 2014. 2

[LY94] Carsten Lund and Mihalis Yannakakis. On the hardness of approximating minimization problems. *J. ACM*, 41(5):960–981, 1994. 3

[Rot11] Thomas Rothvoß. Directed steiner tree and the lasserre hierarchy. *CoRR*, abs/1111.5473, 2011. 2, 13

[Zel97] Alexander Zelikovsky. A series of approximation algorithms for the acyclic directed steiner tree problem. *Algorithmica*, 18(1):99–110, 1997. 3, 6